

Assignment-8

ELP - 718 Telecom Software Laboratory

Sudhanshu Chaudhary

2019JTM2207

2019-2021

A report presented for the assignment on
Python and Github



Bharti School Of
Telecommunication Technology and Management
IIT Delhi

India

September 18, 2019

Contents

1	Problem Statement-1	3
1.1	Problem Statement	3
1.2	Algorithm and Implementation	3
1.3	Program Structure	4
1.4	Screenshots	5
2	Problem Statement-2	6
2.1	Problem Statement	6
2.2	Algorithm and Implementation	7
2.3	Program Structure	8
2.4	Screenshots	8
A	Appendix	9
A.1	Code for ps1	9
A.2	Code for ps2	11

List of Figures

1	Git Clone	5
2	Git status	5
3	Git status	5
4	Ps1	6
5	Git	8
6	Ps2	9

1 Problem Statement-1

1.1 Problem Statement

Parity Check

The simplest way of error detection is to append a single bit, called a parity check, to a string of data bits. This parity check bit has the value 1 if the number of 1's in the bit string is even and has the value 0 otherwise, i.e., Odd Parity Check.

Bit-Oriented Framing

Data Link Layer needs to pack bits into frames so that each frame is distinguishable from another. Frames can be fixed or variable size. In variable size framing, we define the end of the frame using a bit-oriented approach. It uses a special string of bits, called a flag for both idle fills and to indicate the beginning and the ending of frames. The bit stuffing rule is to insert a 0 after each appearance of 010 in the original data. The string 0101 is used as the bit string or flag to indicate the end of the frame.

Input Format

Enter binary bit data that has to be transmitted.

Output Format

Print binary bit data with parity bit.

Print the modified string that is to be transmitted

Sample Input

010101110100101

Sample Output

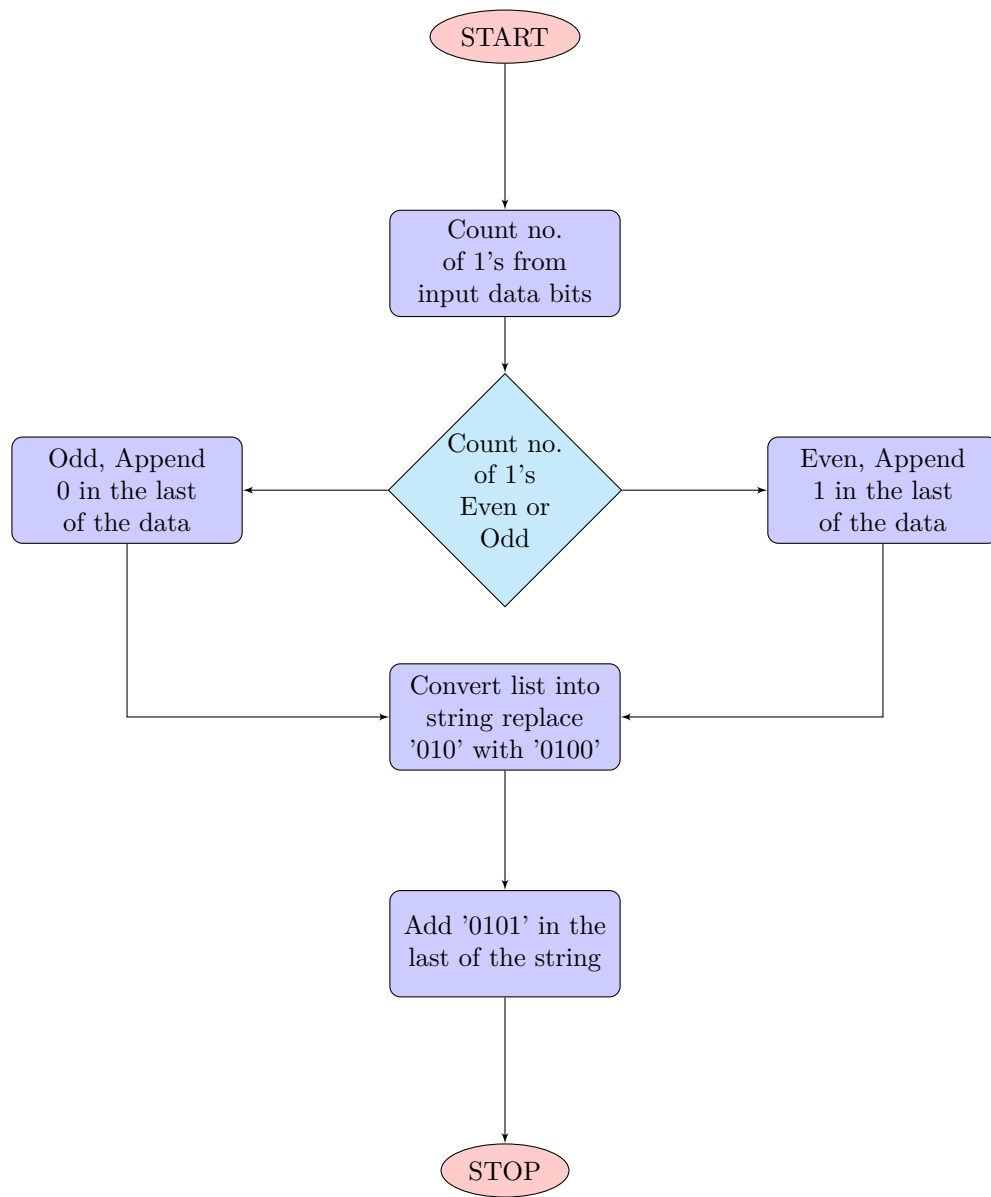
Parity bit data : 0101011101001011

Transmitting data: 01001011101000100110101

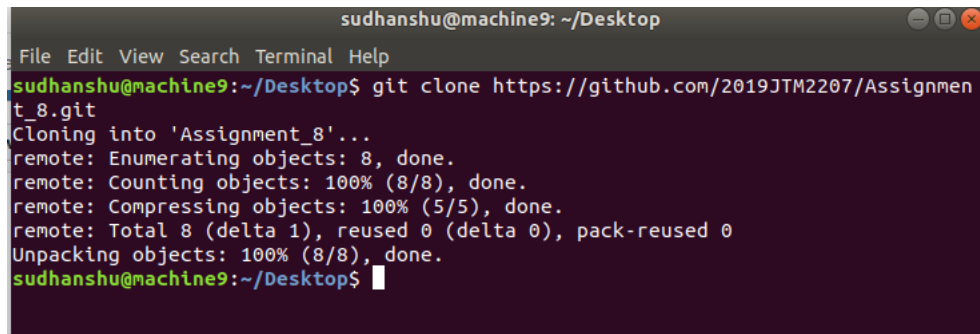
1.2 Algorithm and Implementation

1. Take the input data bits from user.
2. Count number of 1's in the data.
3. If count is even append 1 in the last of the data.
4. If count is odd append 0 in the last of the data.
5. Convert list into string.
6. Replace '010' with '0100' in the new string.
7. Add '0101' in the last of the string.

1.3 Program Structure

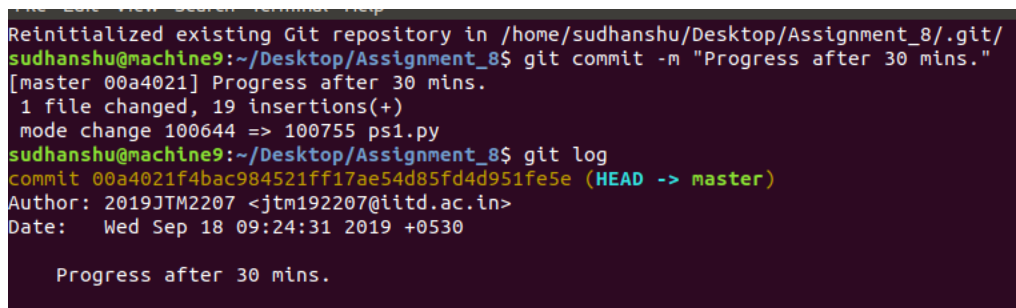


1.4 Screenshots



```
sudhanshu@machine9: ~/Desktop
File Edit View Search Terminal Help
sudhanshu@machine9:~/Desktop$ git clone https://github.com/2019JTM2207/Assignment_8.git
Cloning into 'Assignment_8'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (8/8), done.
sudhanshu@machine9:~/Desktop$
```

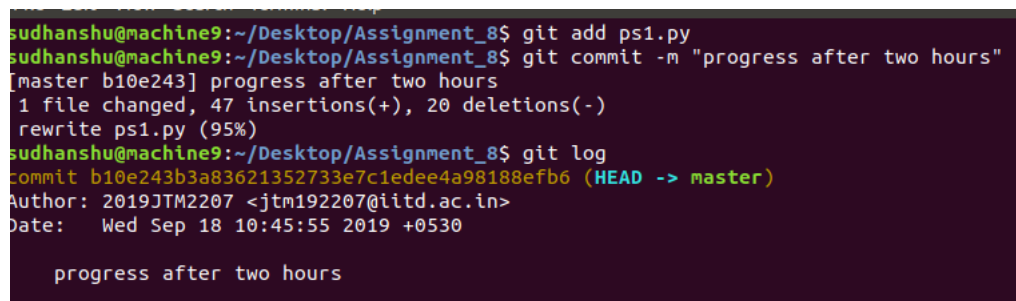
Figure 1: Git Clone



```
Reinitialized existing Git repository in /home/sudhanshu/Desktop/Assignment_8/.git/
sudhanshu@machine9:~/Desktop/Assignment_8$ git commit -m "Progress after 30 mins."
[master 00a4021] Progress after 30 mins.
1 file changed, 19 insertions(+)
mode change 100644 => 100755 ps1.py
sudhanshu@machine9:~/Desktop/Assignment_8$ git log
commit 00a4021f4bac984521ff17ae54d85fd4d951fe5e (HEAD -> master)
Author: 2019JTM2207 <jtm192207@iitd.ac.in>
Date: Wed Sep 18 09:24:31 2019 +0530

Progress after 30 mins.
```

Figure 2: Git status



```
sudhanshu@machine9:~/Desktop/Assignment_8$ git add ps1.py
sudhanshu@machine9:~/Desktop/Assignment_8$ git commit -m "progress after two hours"
[master b10e243] progress after two hours
1 file changed, 47 insertions(+), 20 deletions(-)
rewrite ps1.py (95%)
sudhanshu@machine9:~/Desktop/Assignment_8$ git log
commit b10e243b3a83621352733e7c1edee4a98188efb6 (HEAD -> master)
Author: 2019JTM2207 <jtm192207@iitd.ac.in>
Date: Wed Sep 18 10:45:55 2019 +0530

progress after two hours
```

Figure 3: Git status

```

sudhanshu@machine9: ~/Desktop/Assignment_8
File Edit View Search Terminal Help
sudhanshu@machine9:~/Desktop/Assignment_8$ ./ps1.py
Enter binary bit data that has to be transmitted= 010101110100101
Parity bit data : 0101011101001011
Transmitting data: 01010111010010110101
sudhanshu@machine9:~/Desktop/Assignment_8$ ./ps1.py
Enter binary bit data that has to be transmitted= 0101011101001011
Parity bit data : 01010111010010110
Transmitting data: 010101110100101100101
sudhanshu@machine9:~/Desktop/Assignment_8$

```

Figure 4: Ps1

2 Problem Statement-2

2.1 Problem Statement

3X3 Numeric Tic-Tac-Toe (Use numbers 1 to 9 instead of X's and O's) One player plays with the odd numbers (1, 3, 5, 7, 9) and the other player plays with the even numbers (2,4,6,8). All numbers can be used only once. The player who puts down 15 points in a line wins (sum of 3 numbers). Always Player with odd numbers starts the game. Once a line contains two numbers whose sum is 15 or greater, there is no way to complete that line, although filling in the remaining cells might be necessary to complete a different line.
Note – Line can be horizontal, vertical or diagonal

Constraints:

$1 \leq \text{Position} \leq 9$
 $1 \leq \text{Number} \leq 9$

Terminal:

- Print 'Welcome to the Game!'.
- Print whether it is Player 1's or Player 2's chance.
- Get the position and number to be entered from the user.
- Show tic tac toe with data.
- Continue till the game gets draw or some player wins and show the result.
- Ask the user whether to continue for the next game or exit.

Sample Output:

Welcome to the Game!

Player 1's chance

Enter the position and number to be entered: 5,3

	3	

Player 2's chance Enter the position and number to be entered: 7,4

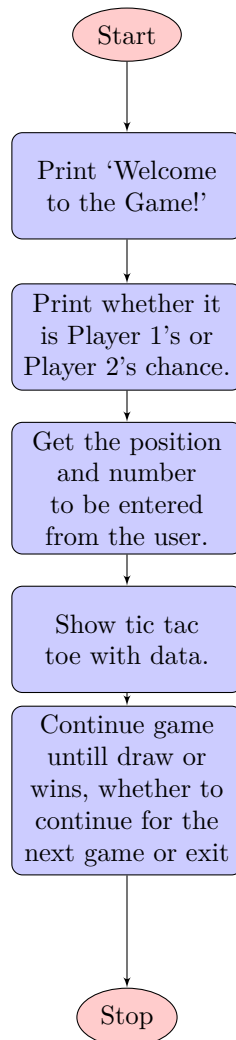
	3	
4		

... Continue till game ends Note – **Must** use at least one User Defined Function.

2.2 Algorithm and Implementation

1. Print ‘Welcome to the Game!’.
2. Print whether it is Player 1’s or Player 2’s chance.
3. Get the position and number to be entered from the user.
4. Show tic tac toe with data.
5. Continue till the game gets draw or some player wins and show the result.
6. Ask the user whether to continue for the next game or exit.

2.3 Program Structure



2.4 Screenshots

```
File Edit View Search Terminal Help
sudhanshu@machine9:~/Desktop/Assignment_8$ git add ps2.py
sudhanshu@machine9:~/Desktop/Assignment_8$ git commit -m "Second problem started"
[master 06117b0] Second problem started
 1 file changed, 42 insertions(+)
 mode change 100644 => 100755 ps2.py
sudhanshu@machine9:~/Desktop/Assignment_8$ git log
commit 06117b022165224143f176519a53c4314f07406b (HEAD -> master)
Author: 2019JTM2207 <jtm192207@iitd.ac.in>
Date: Wed Sep 18 11:16:27 2019 +0530

    Second problem started
```

The screenshot shows a terminal window with the following content: A menu bar with 'File Edit View Search Terminal Help'. The terminal shows the user 'sudhanshu@machine9' in the directory '~/Desktop/Assignment_8'. The commands executed are 'git add ps2.py', 'git commit -m "Second problem started"', and 'git log'. The output of 'git commit' shows a commit to master with 1 file changed and 42 insertions. The output of 'git log' shows the commit details, including the commit hash, author, date, and the commit message 'Second problem started'.

Figure 5: Git

```
File Edit View Search Terminal Help
sudhanshu@machine9:~/Desktop/Assignment_8$ ./ps2.py
Welcome to the Game!
Enter the position : 1
Enter the number : 1
Both position and number are valid
Player 1's chance
 0 0 | | 
--|---|
 0 0 | | 
--|---|
 0 0 | | 
--|---|
 0 0 | | 
--|---|
The count is: 0
Enter the position : █
```

Figure 6: Ps2

A Appendix

A.1 Code for ps1

```
#!/usr/bin/python3

#Python code to get parity

n = (input("Enter binary bit data that has to be transmitted= "))
y = int(n)

#Total number of digits in the input data
count = 0
while y!=0:
    y = round(y/10)
    count = count+1
#print("Total bits in the data ", count)

#Convert string to list

bits = [int(d) for d in str(n)]
#print (bits)

#counting number of 1's in the enter binary bit data that has to be transmitted.

count1 = 0
for i in bits:
    if i == 1:
        count1 = count1+1
    else:
        count = count

#adding parity bit in end of the data
if count1%2 == 0:
```

```

bits.append(1)
#print(bits)
else:
bits.append(0)
#print(bits)

#convert list into string

modifiedstring = ''.join(str(e) for e in bits)
print("Parity bit data :",modifiedstring)

#The string 0101 is used as the bit string or flag to indicate the end of the frame.
m = "0101"
o = modifiedstring+m
print("Transmitting data: ",o)

```

```
#!/usr/bin/python3

board = [' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ']

print("Welcome to the Game!")
player = 0

count = 0
while count < 9:

    print("Enter the position : ",end="")
    position = int(input())
    print("Enter the number : ",end="")
    number = int(input())

    #check for constraints
    if position >=1 and position <=9:
        if number >=1 and number <=9:
            print("Both position and number are valid")
            board[int(position)] = int(number)
        else:
            print("Number is invalid")
        else:
            print("Position is invalid")

    #check for whether it is Player 1's or Player 2's chance.

    if player == 0:
        print("Player 1's chance")
        player = player+1
    else:
        print("Player 2's chance")
        player = player-1


# Function for Draws Game Board
def DrawBoard():
    print(" %c | %c | %c " % (board[1],board[2],board[3]))
    print("___|___|___")
    print(" %c | %c | %c " % (board[4],board[5],board[6]))
    print("___|___|___")
    print(" %c | %c | %c " % (board[7],board[8],board[9]))
    print("   |   |   ")

DrawBoard()

# Function for Checks position is empty or not
def CheckPosition(board):
for i in range(0, 8):
if(board[i] >=1 and board[i] <=9):
```

```

return True
else:
return False
# function for checking sum = 15
def checksum():
if (board[1]+board[2],board[3] == 15):
return 1
elif (board[4]+board[5],board[6] == 15):
return 1
elif (board[9]+board[7],board[8] == 15):
return 1
elif (board[1]+board[4],board[7] == 15):
return 1
elif (board[2]+board[5],board[8] == 15):
return 1
elif (board[3]+board[6],board[9] == 15):
return 1
elif (board[1]+board[5],board[9] == 15):
return 1
elif (board[3]+board[5],board[7] == 15):
return 1
else:
return 0

```

```

print ('The count is:', count)
count = count + 1

```

References

- [1] *Append and extend in Python.* <https://www.geeksforgeeks.org/append-extend-python/>.
- [2] *Convert list to string.* <https://stackoverflow.com/questions/5618878/how-to-convert-list-to-string>.
- [3] *Python Loops.* https://www.w3schools.com/python/python_while_loops.asp.
- [4] *While loop in python.* <https://www.programiz.com/python-programming/while-loop>.